Tshwane University of Technology
We empower people

YEAR: 2016
EXAMINATION: C

May Main

SUBJECT NAME:	DEVELOPMENT SOFTWARE IV
SUBJECT CODE:	DSO401T
QUALIFICATION(S):	BTech: IT: Software Development (BTIS05)

PAPER DESCRIPTION: Computer-based **DURATION:** 4 Hrs **PAPER:** Only

SPECIAL REQUIREMENTS

- ☒ **NONE**
- ☐ **NON-PROGRAMMABLE POCKET CALCULATOR**
- ☐ **SCIENTIFIC CALCULATOR**
- ☐ **COMPUTER ANSWER SHEET**
- ☐ **GRAPH PAPER**
- ☐ **DRAWING INSTRUMENTS**

OTHER: Database sample table description

INSTRUCTIONS TO CANDIDATES: Answer all questions

Submit your question paper to the examiner / invigilator(s) after the examination with your student#, name and PC# clearly written on it.

TOTAL NUMBER OF PAGES INCLUDING COVER PAGE: 5
TOTAL NUMBER OF ANNEXURES:

EXAMINER:	SA Odunaike	**FULL MARKS:**	100
MODERATOR:	A Dandadzi	**TOTAL MARKS:**	100

Question 1 **[40]**

Customer order and item information program comprises of some stored procedures and functions which are used to extract information about customer, their order and order items respectively as shown below and their requirement are specified from 1.1 through 1.4:

1.1 A procedure is required that will accept an order number and later extract the customer information as shown below: (10)

101-TKB SPORT SHOP
490 BOLI RD.
REDWOOD CITY CA (94061)
Telephone: (415) 368-1223
Agent: 7521-WARD

1.2 Another procedure is required that will receive the same order identification number as in 1.1 above and extract the information pertaining to the order as shown below: (10)

```
1.    100860-ACE TENNIS RACKET I          1    35     35
2.    100870-ACE TENNIS BALLS-3 PACK      3    2.8    8.4
3.    100890-ACE TENNIS NET               1    58     58
```

1.3 A function is required that will receive the same order identification number as in 1.1 above and then calculates the total value of the order as shown below: (10)

Sub Total: 101.4

1.4 Write a PL/SQL anonymous block that calls all the procedures and the function as specified in Q1.1 –Q1.3. Make provision to handle invalid data error. (10)

```
Enter value for order_no: 617

Customer details:
105-K + T SPORTS
3476 EL PASEO
SANTA CLARA CA (91003)
Telephone:  (408) 376-9966
Agent: 7844-TURNER

Order details:
Item   Product                            Qty    Price   Total
1.     100860-ACE TENNIS RACKET I         50     35      1750
2.     100861-ACE TENNIS RACKET II        100    45      4500
3.     100870-ACE TENNIS BALLS-3 PACK     500    2.8     1400
4.     100871-ACE TENNIS BALLS-6 PACK     500    5.6     2800
5.     100890-ACE TENNIS NET              500    58      29000
6.     101860-SP TENNIS RACKET            100    24      2400
7.     101863-SP JUNIOR RACKET            200    12.5    2500
8.     102130-RH: "GUIDE TO TENNIS"       100    3.4     340
9.     200376-SB ENERGY BAR-6 PACK        200    2.4     480
10.    200380-SB VITA SNACK-6 PACK        300    4       1200

Sub Total: 46370
Vat: 6491.8
Total: 52861.8
```

Question 2 [15]

Write a PL/SQL package and body called ADD_ORDER that use overloaded procedures to add new order information into the required table. The first definition of the overloaded procedure defines all columns to be provided as parameters to the procedure except the order identification number which is generated from the last known publisher number value. The second definition of the procedure defines all columns to be provided explicitly as parameters to the procedure.

Question 3 [25]

Create a procedure called product order summary report that will accept three parameters (file directory, file name and product identification number) to generate a text file report of all orders per given product. Make provision to handle errors of invalid file handling and write error (application error -20001 and -20002 respectively) resulting from the use of UTF_FILE package.

PRODUCT SUMMARY SALES REPORT:

GENERATED ON [today's date]

Product code: 102130
Product Description: RH: "GUIDE TO TENNIS"

Product and order summary:

Order#	Product	Qty	Item Total	Order Total
605	106-SHAPE UP	10	34	8324
606	100-JOCKSPORTS	1	3.4	3.4
616	103-JUST TENNIS	10	34	764
617	105-K + T SPORTS	100	340	46370
619	104-EVERY MOUNTAIN	100	340	1260
620	100-JOCKSPORTS	500	1700	4450

Product ordered in 6 order(s)
No of Product ordered: 721
Total Amount spends on the product:R2451.4
Total Budget for all orders: R61171.4

***** End of report *****

Question 4 [20]

4.1 Write a trigger called **CHECK_ORDER** that validates the addition of a new order or modification of the order number or shipment date. The trigger should raise relevant application error messages in the following instances: (15)

- The shipment date must be at least one week later than the order date
- The order number may not be changed
- May not add a duplicate order to the database
- A new order has been added

4.2 Write a trigger called **CHECK_PRICE** to protect the data integrity of changes to the standard price. Ensure that the standard price will always be more than the minimum cost price but not more than 50% increase of the minimum costing price. Meaning, if the previous minimum cost price is R100, the new standard selling price may not be R100 or less and not more than R150 otherwise the trigger must raise an appropriate application error message. (5)

Tshwane University of Technology

We empower people

MEMORANDUM

CONFIDENTIAL

A ☐	B ☐	C ☒
Distance Education Exam ☐		

SUBJECT CODE:	DSO401T
SUBJECT NAME:	DEVELOPMENT SOFTWARE IV
EXAMINATION DATE: (For Office Use Only)	29X

Contact person(s) to collect the scripts:

	Examiner	Moderator
Name	SA Odunaike	A Dandadzi
Campus (If Applicable)	Soshanguve	
Office Address	20-112	Dept of Computer Science University of Limpopo Medunza Campus
Work Tel No.	0123829151	
Mobile No.	0826774963	0822022134

For Office use only:

Applicable Campus	(x)	No.
ARCADIA	☐	
ARTS	☐	
EMALAHLENI	☐	
GA-RANKUWA	☐	
NELSPRUIT	☐	
POLOKWANE	☐	
PRETORIA	☐	
SOSHANGUVE	☒	
DISTANCE EDUCATION UNIT	☐	
EXTRAS	☐	
TOTAL COPIES		

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Tshwane University of Technology *We empower people*	**DEVELOPMENT SOFTWARE IV** ~~**DSO401T**~~	
	Examination C memo **June, 2016** **Full Marks: 100**	**Examiner:** SA Odunaike **Moderator:** A Dandadzi **PC No.:**
	Student number:	
	Surname	**Initials** — **%**

QUESTION	1	2	3	4	MAX	Signatures
TOTAL MARK	40	15	25	20	100	
EXAMINER MARK						
STUDENT MARK						

GENERAL INSTRUCTIONS:

- Read the ***program requirement and question*** carefully before attempting to formulate a solution.
- Create a PL/SQL block to answer the questions that follows.
- Use Kaleidoscope database and carefully study the sample database
- All output must be displayed by using the DBMS package built-in.
- Remember to SET SERVEROUTPUT ON in order to display the output generated from your PL/SQL block.
- Type / create your program using notepad only (one notepad only), states the question number being answered clearly
- Save your file as **StudentNumber** and upload the text file only

Question 1 [40]

Customer order and item information program comprises of some stored procedures and functions which are used to extract information about customer, their order and order items respectively as shown below and their requirement are specified from 1.1 through 1.4:

1.1 A procedure is required that will accept an order number and later extract the customer information as shown below: (10)

101-TKB SPORT SHOP
490 BOLI RD.
REDWOOD CITY CA (94061)
Telephone: (415) 368-1223
Agent: 7521-WARD

```
CREATE OR REPLACE PROCEDURE cust_info √
(pordid     IN     number,
pcustid     OUT    number,
pname       OUT    varchar2,
padd        OUT    varchar2,
pcty        OUT    varchar2,
psta        OUT    varchar2,
pzip        OUT    varchar2,
parea       OUT    varchar2,
pphone      OUT    varchar2,
prepid      OUT    number,
prep        OUT    varchar2)

IS

BEGIN

SELECT a.custid, name, address, city, state, zip, area, phone, repid, ename √√
INTO pcustid, pname, padd, pcty, psta, pzip, parea, pphone, prepid, prep
FROM o_ord a, o_customer b, o_emp c √√
WHERE a.custid = b.custid √
AND b.repid =c.empno √√
AND a.ordid = pordid; √

END cust_info;
/
```

1.2 Another procedure is required that will receive the same order identification number as in 1.1 above and extract the information pertaining to the order as shown below: (10)

```
1.   100860-ACE TENNIS RACKET I        1    35    35
2.   100870-ACE TENNIS BALLS-3 PACK    3    2.8   8.4
3.   100890-ACE TENNIS NET             1    58    58
```

```
CREATE OR REPLACE PROCEDURE ord_info √
(pord          IN number)
IS

CURSOR o_info is
SELECT itemid, a.prodid, descrip, qty, actualprice, itemtot √√√
FROM o_item a, o_product b √
WHERE a.prodid = b.prodid  √
AND a.ordid = pord; √

BEGIN

FOR o_rec IN o_info√

LOOP

DBMS_OUTPUT.PUT_LINE(o_rec.itemid||'. '||o_rec.prodid||'-'||o_rec.descrip||'  '||o_rec.qty
||'  '||o_rec.actualprice||'  '||o_rec.itemtot); √√

END LOOP;
END ord_info;
/
```

1.3 A function is required that will receive the same order identification number as in 1.1 above and then calculates the total value of the order as shown below: (10)

Sub Total: 101.4

```
CREATE OR REPLACE FUNCTION calc_ord  √
(pord    IN        number)

RETURN NUMBER √
IS
v_tot   number(8,2) := 0; √

BEGIN
SELECT sum(itemtot)  √√√
INTO v_tot√
FROM o_item    √
WHERE ordid = pord;  √

RETURN v_tot;  √
END calc_ord;
/
```

1.4 Write a PL/SQL anonymous block that calls all the procedures and the function as specified in Q1.1 –Q1.3. Make provision to handle invalid data error. (10)

```
Enter value for order_no: 617
Customer details:
105-K + T SPORTS
3476 EL PASEO
SANTA CLARA CA (91003)
Telephone:  (408) 376-9966
Agent: 7844-TURNER
Order details:
Item    Product                              Qty    Price   Total
1.      100860-ACE TENNIS RACKET I           50     35      1750
2.      100861-ACE TENNIS RACKET II          100    45      4500
3.      100870-ACE TENNIS BALLS-3 PACK       500    2.8     1400
4.      100871-ACE TENNIS BALLS-6 PACK       500    5.6     2800
5.      100890-ACE TENNIS NET                500    58      29000
6.      101860-SP TENNIS RACKET              100    24      2400
7.      101863-SP JUNIOR RACKET              200    12.5    2500
8.      102130-RH: "GUIDE TO TENNIS"         100    3.4     340
9.      200376-SB ENERGY BAR-6 PACK          200    2.4     480
10.     200380-SB VITA SNACK-6 PACK          300    4       1200
Sub Total: 46370
Vat: 6491.8
Total: 52861.8
```

```
DECLARE
vordid          number(4):=&order_no;  √
vcustid         number(3);
vname           varchar2(45);
vadd            varchar2(25);
vcty            varchar2(15);
vsta            varchar2(15);
vzip            varchar2(6);
varea           number(3);
vphone          varchar2(8);
vrepid          number(4);
vrep            varchar2(10);
ptot            number(9,2):=0;
Vat             number(8,2):=0;
tamt            number(9,2):=0;

BEGIN
cust_info(vordid, vcustid, vname, vadd, vcty, vsta, vzip, varea, vphone, vrepid, vrep); √√
DBMS_OUTPUT.PUT_LINE('Customer details: ');
DBMS_OUTPUT.PUT_LINE(vcustid||'-'||vname);
DBMS_OUTPUT.PUT_LINE(vadd);
DBMS_OUTPUT.PUT_LINE(vcty||' '||vsta||' ('||vzip||')');          √
DBMS_OUTPUT.PUT_LINE('Telephone: '||' ('||varea||') '||vphone);
DBMS_OUTPUT.PUT_LINE('Agent: '||vrepid||'-'||vrep);
DBMS_OUTPUT.PUT_LINE('Order details: ');

ord_info(vordid);  √
ptot:=calc_ord(vordid); √

vat := (14/100) * ptot;    √
tamt := ptot + vat; √

DBMS_OUTPUT.PUT_LINE('Sub Total: '||ptot);
DBMS_OUTPUT.PUT_LINE('Vat: '||vat);                 √
DBMS_OUTPUT.PUT_LINE('Total: '||tamt);
EXCEPTION
WHEN no_data_found THEN
DBMS_OUTPUT.PUT_LINE('Invalid order number / Re-enter:'); √
END;
/
```

Question 2 **[15]**

Write a PL/SQL package and body called ADD_ORDER that uses overloaded procedures to add new order information into the required table. The first definition of the overloaded procedure defines all columns to be provided as parameters to the procedure except the order identification number which is generated from the last known publisher number value. The second definition of the procedure defines all columns to be provided explicitly as parameters to the procedure.

```
CREATE OR REPLACE PACKAGE add_order √
IS
PROCEDURE new_ord √
(porderd          IN     date,
pcomp             IN     varchar2,
pcustid           IN     varchar2, √
pshipd            IN     date,
ptotal            IN     number)

PROCEDURE new_ord √
(pordid           IN     number,
porderd           IN     date,
pcomp             IN     varchar2,
pcustid           IN     varchar2, √
pshipd            IN     date,
ptotal            IN     number))
END add_order;
/
```

```
CREATE OR REPLACE PACKAGE BODY add_order IS √
PROCEDURE new_ord
(porderd            IN      date,
pcomp               IN      varchar2,
pcustid             IN      varchar2, √
pshipd              IN      date,
ptotal              IN      number)
IS
BEGIN
INSERT INTO O_ORD (ORDID, ORDERDATE, COMMPLAN, CUSTID, SHIPDATE, TOTAL) √
VALUES ((SELECT max(ordid)+1 FROM O_ORD) √√, porderd, pcomp, pcustid, pshipd, ptotal);
END new_ord;

PROCEDURE new_ord
(pordid             IN      number,
porderd             IN      date,
pcomp               IN      varchar2,
pcustid             IN      varchar2, √
pshipd              IN      date,
ptotal              IN      number)
IS
BEGIN
INSERT INTO O_ORD (ORDID, ORDERDATE, COMMPLAN, CUSTID, SHIPDATE, TOTAL) √
VALUES ((SELECT max(ordid)+1 FROM O_ORD) √√, porderd, pcomp, pcustid, pshipd, ptotal); √
END new_ord;
END add_order;
/
```

Question 3 [25]

Create a procedure called product order summary report that will accept three parameters (file directory, file name and product identification number) to generate a text file report of all orders per given product. Make provision to handle errors of invalid file handling and write error (application error -20001 and -20002 respectively) resulting from the use of UTF_FILE package.

```
PRODUCT SUMMARY SALES REPORT:

GENERATED ON [today's date]

Product code: 102130
Product Description: RH: "GUIDE TO TENNIS"

Product and order summary:
Order#      Product                 Qty    Item Total     Order Total
605         106-SHAPE UP            10     34             8324
606         100-JOCKSPORTS          1      3.4            3.4
616         103-JUST TENNIS         10     34             764
617         105-K + T SPORTS        100    340            46370
619         104-EVERY MOUNTAIN      100    340            1260
620         100-JOCKSPORTS          500    1700           4450

Product ordered in 6 order(s)
No of Product ordered: 721
Total Amount spends on the product:R2451.4
Total Budget for all orders: R61171.4

*** End of report ***
```

```
CREATE OR REPLACE PROCEDURE prodord_info √
(p_prodid       IN OUT number,
p_filedir       IN VARCHAR2,          √√
p_filename      IN VARCHAR2)
IS

v_filehandle UTL_FILE.FILE_TYPE; √

CURSOR ord_info is
SELECT a.prodid, descrip, b.ordid,  c.custid||'-'||name customer, qty, itemtot, total √√
FROM o_product a, o_item b, o_customer c, o_ord d √
WHERE a.prodid = b.prodid
AND c.custid = d.custid          √√
AND b.ordid = d.ordid
AND b.prodid = &p_prodid
ORDER BY b.ordid; √

v_cnt           number(2):=0;
v_qty           number(2):=0;   √
v_amt           number(5,2):=0;
v_tot           number(5,2):=0;

BEGIN

v_filehandle := UTL_FILE.FOPEN(p_filedir, p_filename, 'w'); √
UTL_FILE.PUTF (v_filehandle, PRODUCT SUMMARY SALES REPORT:'); √
UTL_FILE.PUTF (v_filehandle, 'GENERATED ON %s\n', SYSDATE);
UTL_FILE.NEW_LINE(v_filehandle);

FOR ord_rec IN ord_info√

UTL_FILE.PUTF (v_filehandle, 'Product code: %s\n', ord_rec.prodid); √
UTL_FILE.PUTF (v_filehandle, 'Product Description: %s\n', ord_rec.descrip); √
UTL_FILE.NEW_LINE(v_filehandle);
UTL_FILE.PUTF (v_filehandle,' Product and order summary:');
UTL_FILE.PUTF (v_filehandle,' Order#  Product         Qty     Item Total       Order Total '); √

LOOP

UTL_FILE.PUTF (v_filehandle,' %s %s %s %s %s \n', ord_rec.ordid, ord_rec.customer,
ord_rec.qty, ord_rec.itemtot, ord_rec.total); √√

v_cnt := v_cnt + 1;
v_qty := v_qty + ord_rec.qty;
v_amt := v_amt + ord_rec.itemtot; √√
v_tot:= v_tot + ord_rec.v_total;

END LOOP;
```

```
UTL_FILE.PUTF (v_filehandle, 'Product ordered in %s order(s)\n', v_cnt);
UTL_FILE.PUTF (v_filehandle, 'No of Product ordered: %s\n', v_qty);
UTL_FILE.PUTF (v_filehandle, 'Total Amount spends on the product: R%s\n', v_amt); √√
UTL_FILE.PUTF (v_filehandle, 'Total Budget for all orders: R%s\n', v_tot);
UTL_FILE.PUT_LINE (v_filehandle, '*** END OF REPORT ***');
UTL_FILE.FCLOSE (v_filehandle);

EXCEPTION

WHEN UTL_FILE.INVALID_FILEHANDLE THEN
RAISE_APPLICATION_ERROR (-20001, 'Invalid File.'); √
WHEN UTL_FILE.WRITE_ERROR THEN
RAISE_APPLICATION_ERROR (-20002, 'Unable to write to file'); √
END prodord_Info;
/
```

Question 4 [20]

4.1 Write a trigger called **CHECK_ORDER** that validates the addition of a new order or modification of the order number or shipment date. The trigger should raise relevant application error messages in the following instances: (15)

- The shipment date must be at least one week later than the order date
- The order number may not be changed
- May not add a duplicate order to the database
- A new order has been added

```
CREATE OR REPLACE TRIGGER check_order √
BEFORE UPDATE OR INSERT ON o_ord √
FOR EACH ROW √
DECLARE
        v_order         o_ord.ordid%TYPE;

BEGIN
        IF updating ('shipdate') THEN /* we do not SELECT INTO in updating mode*/√
                IF :NEW.shipdate < :OLD.orderdate+7 THEN √
                        RAISE_APPLICATION_ERROR('-20101',
                        'Shipment date must be at least a week later than order date');     √
                END IF;
        ELSIF updating ('ordid') THEN √
                IF :NEW.ordid <> :OLD.ordid THEN √
                        RAISE_APPLICATION_ERROR('-20102',
                        'Order number may not be changed'); √
                END IF;
        END IF;

        IF inserting THEN       √
                SELECT ordid
                        INTO v_order
                        FROM o_ord
                        WHERE ordid = :NEW.ordid; √√
                RAISE_APPLICATION_ERROR('-20103',
                'May not add a duplicate order to the database'); √
        END IF;
        EXCEPTION WHEN no_data_found THEN  /* order does not exist – add */√
         DBMS_OUTPUT.PUT_LINE(' New order number '||:NEW.order#||' added'); √
END;
/
```

4.2 Write a trigger called **CHECK_PRICE** to protect the data integrity of changes to the standard price. Ensure that the standard price will always be more than the minimum cost price but not more than 50% increase of the minimum costing price. Meaning, if the previous minimum cost price is R100, the new standard selling price may not be R100 or less and not more than R150 otherwise the trigger must raise an appropriate application error message. (5)

```
CREATE OR REPLACE TRIGGER check_price √
BEFORE UPDATE OR INSERT OF retail ON o_price     √
FOR EACH ROW
WHEN (:NEW.stdprice <= :NEW.minprice) and (:NEW.stdprice > 1.5*:NEW.minprice) √
BEGIN
  RAISE_APPLICATION_ERROR('-20101',
  'Retail increase may not be less than minimum price and not more than 50% increase of the
minimum price');  √√
END;
/
```